Amendments to the Claims:

This Listing of Claims will replace all prior versions, and listings, of claims in the application. Compared to prior versions, claims 1-4 and 7-22 are presented for examination. Claims 1-3, 7, 8, 13, 14 and 16-22 are amended while 4, 9-12 and 15 are original or as previously presented. Claims 5 and 6 are canceled.

Listing of Claims:

1. (Currently Amended) A method for validating executable code resident in an operating system having executable instructions, comprising the steps of:

identifying an executable code having an unaltered size;

calculating an initial score associated with the executable code when the executable code is initially or shortly thereafter loaded into an operating system;

saving the initial score;

calculating a plurality of subsequent scores on the executable code;

exclusively comparing each of the subsequent scores to the saved initial score and to no other scores;

if the each of the subsequent scores ~~does~~ do not vary from the saved initial score, concluding the executable code maintains the unaltered size; and

if any of the subsequent scores ~~varies~~ vary from the saved score, concluding the executable code has an altered size.

2. (Currently Amended) The method of claim 1, further comprising the steps of:

unloading the executable code from the operating system if the saved initial score is not equal to the any of the subsequent scores.

3. (Currently Amended) The method of claim 1, further comprising the steps of:

disabling at least a portion of the executable code if the saved initial score is not equal to the any of the subsequent scores.

4. (Original) The method of claim 1, wherein the scores are the result of a checksum calculation.

5. (Canceled)

6. (Canceled)

7. (Currently Amended) The method of claim 1, further comprising the steps of:

notifying electronically an owner of the executable code if the saved initial score is not equal to the any of the subsequent scores.

8. (Currently Amended) A method for disabling executable code which has been modified without authorization having executable instructions, comprising the steps of:

identifying an executable code in an operating system having an unaltered format;

calculating a score associated with the executable code exclusively within an operating system of a computing device independent of a system management mode of operation;

calculating ~~one or more~~ subsequent scores associated with the executable code;

determining the executable code has an altered format if the score is not equal to any of the subsequent scores, the determining exclusively including comparing each of the subsequent scores to the score with no other score comparisons occurring; and

disabling the executable code if the score is not equal to any of the subsequent scores.

9. (Original) The method of claim 8, further comprising the steps of:

notifying an owner of the executable code if disabled.

10. (Original) The method of claim 8, wherein the scores are the result of a checksum calculation.

11. (Previously Presented) The method of claim 8, wherein the subsequent scores are calculated randomly.

12. (Previously Presented) The method of claim 8, wherein the subsequent scores are calculated at one or more predetermined time intervals.

13. (Currently Amended) The method of claim 8, further comprising the steps of:

removing the executable code if disabled from a memory of ~~an~~ the operating system wherein the executable code resides.

14. (Currently Amended) The method of claim 8, further comprising the steps of:

assisting in the loading of the executable code, if not disabled, to a memory of ~~an~~ the operating system wherein the executable code resides.

15. (Original) The method of claim 8, further comprising the steps of:

registering the executable code if not disabled; and recording a history if the executable code is disabled.

16. (Currently Amended) A method of authenticating executable code resident in a memory having executable instructions, comprising the steps of:

identifying an executable code having an unaltered format;

acquiring a score associated with an executable code which was established when the executable code was first loaded into a memory of an operating system;

receiving [[a]] subsequent scores on the executable code while the executable code is in the memory;

exclusively comparing the subsequent scores to the score with neither the subsequent scores nor the score being compared to any other values;

if the subsequent scores ~~does~~ do not vary from the score, executing the executable code in the unaltered format; and

if any of the subsequent scores ~~varies~~ vary from the saved score, indicating the executable code has an altered format.

17. (Currently Amended) The method of claim 16, further comprising the steps of:

disabling the executable code while the executable code is in the memory when the any of the subsequent scores is not equal to the score.

18. (Currently Amended) The method of claim 16, further comprising the steps of:

suspending one or more operations of the executable code while the executable code is executing in the memory when the any of the subsequent scores is not equal to the score.

19. (Currently Amended) The method of claim 16, wherein the subsequent scores [[is]] are received each time the executable code is initiated in the memory for an execution.

20. (Currently Amended) The method of claim 16, reporting one or more system events and variables when the any of the subsequent scores is not equal to the score.

21. (Currently Amended) Functional data used to validate executable code embodied in a computer readable medium, the data comprising:

an unaltered original format;

a first score associated with an executable code when the executable code is initially loaded into an operating system;

a second score associated with the executable code at a period of time subsequent to when the executable code was initially loaded and operable to be exclusively compared with the first score to determine if the executable code has been altered since the initial load; and

an altered unoriginal format if the first and second scores do not equal.

22. (Currently Amended) A system for validating executable code, comprising:

an executable code having an unaltered size;

a scoring set of executable instructions operable to receive and record a score associated with the executable code when the code is initially loaded into a computer readable medium;

a comparing set of executable instructions in an operating system of a computing device independent of a system management mode of operation operable to receive [[a]] subsequent scores associated with the code and to exclusively compare the score and the subsequent scores to determine if the code has been altered; and

an executable code having an altered size if the score and the subsequent scores do not equal.